Beware of digits before the redirection operator

devblogs.microsoft.com/oldnewthing/20060517-00

May 17, 2006

Raymond Chen

If you want to put the string “Meet at 2” into the file “schedule”, you might be tempted to use

```
echo Meet at 2>schedule
```

If you try this, however, you’ll see the string “Meet at” on the screen and the “schedule” file will be blank. [Typo fixed, 10am]

What happened?

A digit immediately before a redirection operator modifies which stream the redirection operator applies to. If you’re going to redirected an alternate output stream, it’ll nearly always be the standard error stream, or stream 2. To put the error output into a file, you would write something like this:

```
sort /invalidswitch 2>errorfile
```

There is also the operator “>&” that reopens a stream as another stream. The idiom

```
some-command >output 2>&1
```

says, “Put the normal output into the file `output`, and then change the error output stream (2) to refer to the normal output stream (1).” The result is that both the regular output and error output end up in the `output` file.

But what if you really want to put the string “Meet at 2” into the file “schedule”?

You can insert a space between the “2” and the “>”. This works for most programs since they ignore trailing spaces on their command line, but this was a trick question: The `echo` command is one of the few commands that actually pays attention to trailing spaces. As a result, the contents of the “schedule” file is “Meet at 2<space><cr><lf>”. Maybe this is close enough for you, in which case you can skip the next paragraph.

But what if you don’t want that trailing space? For that, you can use the metacharacter escape character, the ^:

```
echo Meet at ^2>schedule
```

The last gotcha is that the pesky "2" might come from environment variable expansion.

```
set message=Meet at 2
echo %message%>schedule
```

The trailing "2" in `%message%` interacts with the greater-than sign, leading to an unintended redirection. For this, you can insert a space before the greater-than sign, assuming you are in a scenario where that space is not going to cause you any problems. (And if you're in a scenario where that space will cause a problem, you can use a trick we'll look at next time.)

Mind you, if you're going to take an environment variable whose contents you do not control and expand it onto your command line unquoted, you have much worse problems than a trailing digit messing up your file redirection. Somebody might have decided that the message should be "`&format C: /y`". Inserting this into the command line unquoted would yield "`echo &format C: /y>schedule`" which is a pretty good way to ruin somebody's day. (Well, okay, you can't format a drive with an active pagefile, but you get the idea.)

Raymond Chen

Follow